AltiAgent ActiveX Programmer's Guide

This document provides instructions on configuring and using ActiveX controls with AltiAgent. API reference information is also provided.

Contents

Using an AltiAgent ActiveX Control Object 2
- Understanding Key Files 2
- Basic Procedure to Use an ActiveX Object with AltiAgent 2
- Using AppWizard to Create a Project 3
- Inserting AltiCtrl into dialog 3
- Handle AltiCtrl Events 3
- AltiCtrl Methods 3

AltiAgent ActiveX Control API Reference 4
- StartAltiCtrl 4
- AltiCtrl Event 4
- CallNotify 4

AltiCtrl Methods 7
- SetUserData 7
- GetUserData 7
- SetIVRData 8
- GetIVRData 9
- SetURLData 10
- GetURLData 10
- SetCallerName 11
- GetExtNumber 12
- MakeCall 12
- GetTrunkAccessCode 13

Related Documentation

- AltiGen *AltiAgent Manual* - information on installation and configuration of the AltiAgent client system software.

02/2008

Using an AltiAgent ActiveX Control Object

The AltiAgent ActiveX Control Object is an ActiveX Object. It will work with AltiAgent, getting call-related information from AltiAgent, which acts as a server, and implement actions based on the call information, for example, popping up corresponding customer information and logging caller ID into a database.

This document contains descriptions of AltiAgent ActiveX Control Object files and type library, followed by examples in VC++ and VB.

Understanding Key Files

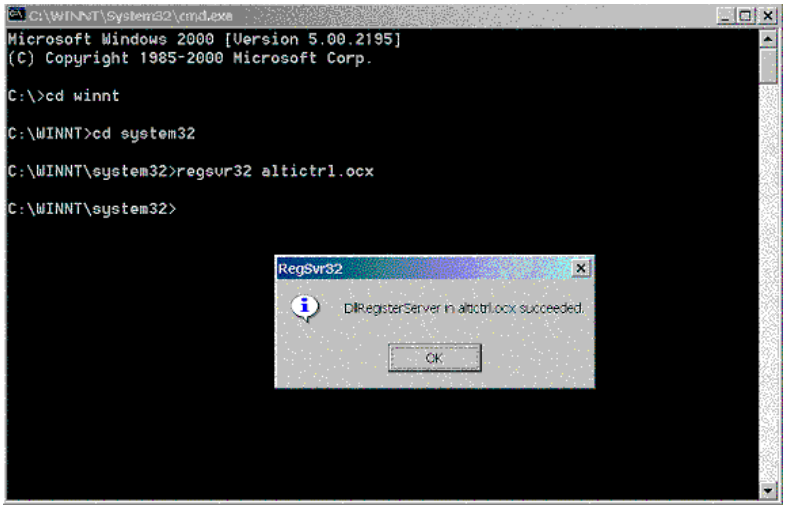
- **AltiCtrl.tlb**—this type library file contains functions within the AltiCtrl ActiveX Object. Methods contained in this type library need to be used directly by Visual C++ developers. In Visual Basic, type library functions are handled automatically at run time.
- **AltiCtrl.ocx**—third party applications use this ActiveX control module to develop applications. It needs to be registered before starting application development.

Note: Before using AltiAgent AltiCtrl to develop your application, you need to register AltiCtrl.ocx using RegSvr32.exe first. Because this object will work with AltiAgent, you need to run an AltiAgent client when you develop and run your application on the same machine.

Basic Procedure to Use an ActiveX Object with AltiAgent

To use ActiveX control object with AltiAgent:

1. Make sure AltiAgent is installed on the client desktop PC.
2. Copy AltiCtrl.ocx file from the AltiAgent API directory to the Windows system32 directory.
3. Bring up the command prompt and register altictrl.ocx



4. Edit VB or VC sample code to interface with destination client application.
5. Compile customized code to an executable file.
6. Copy that executable file to the same directory as AltiAgent.

Using AppWizard to Create a Project

Use AppWizard to create a dialog based MFC project:

1. Choose **MFC AppWizard** (exe) and create a new win32 project named "**ExampleVc**"; press **Next**.
2. Set **Dialog based** as the type of the Application; press **Next**.
3. Keep **ActiveX Control** and **Windows Sockets** checkboxes checked.
4. The program uses shared MFC DLLs (AltiCtrl uses shared MFC DLLs).
5. Click **Finished** to create the project.

Inserting AltiCtrl into dialog

AltiCtrl is an ActiveX Control without any GUI. It is invisible and can be inserted anywhere.

1. Open the main dialog (IDD_EXAMPLEVC_DIALOG)
2. Right-click the dialog and a context menu will display.
3. Choose **Insert ActiveX Control** to display the **Insert ActiveX Control** dialog box.
4. Select **AltiCtrl Control** and click **OK** to add it.
5. Place the control at a suitable location and size it as desired. Please note that the control is invisible at runtime.
6. Identify the control as **IDC_ALTICTRLCTRL1** (default value).

Handle AltiCtrl Events

To add an event process function:

1. Open **Class Wizard** and go to **Message Maps** page.
2. Set **CExampleVcDlg** as current class.
3. Set **IDC_ALTICTRLCTRL1** as current selection of "*Object Ids*" list box and all AltiCtrl Events will be displayed in **Message** list box.
4. Add member functions for the events you would like to handle.
5. Add code for these member functions.

AltiCtrl Methods

When adding a function to handle an AltiCtrl Event, you don't need to worry about how and when the function is called. AltiCtrl will perform the event call back for you.

When using the AltiCtrl properties of getting and setting methods, you can use AltiCtrl to set or monitor a Call property. Before doing so, you must add a member variable associated with this control.

To add a member variable associated with this control:

1. Open **Class Wizard** and go to **Member Variables** page.
2. Set **CExampleVcDlg** as current class.
3. Double-click **IDC_ALTICTRLCTRL1** of **Control Ids** list box and a message box will be displayed to prompt you to add the ActiveX Control **AltiCtrl** into the project. Click **OK** to have **Developer Studio** generate a C++ wrapper class.
4. When the **Confirm Classes** dialog box appears, set **CAltiCtrl** as the class name and click **OK**.

5. In the **Add Member Variable** dialog, create a member variable named **m_altictrl**.
6. Click **OK** to add it to the class. **m_altictrl** is now a member of **CAltiCtrl**.

Now, VC has created a member in class **CExampleVcDlg** which can be used as a general object.

AltiAgent ActiveX Control API Reference

StartAltiCtrl

Description

Start AltiCtrl on the client. This is the first step to use this AltiCtrl.

VC++ Syntax

```
BOOL StartAltiCtrl()
```

VB++ Syntax

```
object.StartAltiCtrl()
```

Return type

BOOL - TRUE if connect to AltiAgent successfully; otherwise FALSE.

Remark

This function is the key of using AltiCtrl. Client cannot call the method or get any event from AltiAgent if this function call fails.

VC++ Example

```
BOOL bstart= m_altictrl.StartAltiCtrl();
```

VB Example

```
Dim bstart as bool
bstart = AltiCtrl1.StartAltiCtrl()
```

AltiCtrl Event

CallNotify

Description

When AltiView/AltiAgent 5.0 receives a call event from AltiServ, AltiCtrl will send the event to your application, each call will have an unique session ID, each call may several events with its unique session ID.

Syntax

```
OnCallNotifyXXXXX (long ulSessionID, short uStatus, BSTR
szCallerID, BSTR szCallerName, BSTR szCallDNIS, BSTR szWorkgroup,
BSTR pszDateTime, BSTR pszDuration)
```

Parameters

ulSessionID	= *Session ID*
uStatus	= *Call Status*
szCallerID	= *Caller ID*
szCallerName	= *Caller Name*
szCallDNIS	= *DNIS*
pszDateTime	= *Call start date and time*
pszDuration	= *Conversation duration*

Remark

Call status value:

LINECALLSTATE_IDLE	0
LINECALLSTATE_OFFERING	1
LINECALLSTATE_ACCEPTED	2
LINECALLSTATE_DIALTONE	3
LINECALLSTATE_DIALING	4
LINECALLSTATE_RINGBACK	5
LINECALLSTATE_BUSY	6
LINECALLSTATE_SPECIALINFO	7
LINECALLSTATE_CONNECTED	8
LINECALLSTATE_PROCEEDING	9
LINECALLSTATE_ONHOLD	10
LINECALLSTATE_CONFERENCED	11
LINECALLSTATE_ONHOLDPENDCONF	12
LINECALLSTATE_ONHOLDPENDTRANSFER	13
LINECALLSTATE_DISCONNECTED	14
LINECALLSTATE_AA	15
LINECALLSTATE_VM	16
LINECALLSTATE_PARK	17
LINECALLSTATE_UNKNOWN	18
LINECALLSTATE_MUSICONHOLD	19
LINECALLSTATE_PLAYVOICEMAIL	20
LINECALLSTATE_CALLPENDING	21
LINECALLSTATE_MONITORSIRENT	22
LINECALLSTATE_CALLBARGEIN	23
LINECALLSTATE_XFER_RINGBACK	24
LINECALLSTATE_XFER_WAITFORONHOOK	25
LINECALLSTATE_DUMMYRING	26
LINECALLSTATE_FORWARDRING	27
LINECALLSTATE_DATALOAD	28
LINECALLSTATE_RECORD	29
LINECALLSTATE_APC	30
LINECALLSTATE_SUPERVISE_COACH	31
LINECALLSTATE_LINE_PARK	32
LINECALLSTATE_RMT_OFFLINE	33

AltiCtrl Methods

SetUserData

Description

Set user data for existing call. User data is used for AltiAgent to describe a call information, for example "This call is urgent" and so on. It can be any information defined by user.

VC++ Syntax

```
BOOL SetUserData(int SessionID , BSTR string)
```

VB Syntax

```
object.SetUserData(SessionID,string)
```

Parameters

SessionID [In] - unique ID for a call

String [In] - set user data for a call; maximum length is 256 bytes. It is only used for trunk line calls.

Return Type

BOOL - if successful TRUE; otherwise FALSE

Remark

SessionID - Session ID received from OnCallNotify event needs to be provided as input to this function. This Session ID is valid in ring event and connect event. After Disconnect event, it is no longer valid.

VC++ Example

```
CComBSTR bstrdata = _T("user data");
m_altictrl. SetUserData (ulSessionID, bstrData);
```

VB Example

```
Dim data As String
Data = "user data"
If (AltiCtrl1.SetUserData(ulSessionID, data) = False) Then
...
Else
...
End If
```

GetUserData

Description

Get user data for existing call.

VC++ Syntax

```
BSTR GetUserData(int SessionID)
```

VB Syntax

```
object.GetUserData(SessionID)
```

Parameters

SessionID - [In]: unique ID for a call

Return Type

String user data for a call; maximum length is 256 bytes.

Remark

SessionID - Session ID received from OnCallNotify event needs to be provided as input to this function. This Session ID is valid in ring event and connect event. After Disconnect event, it is no longer valid.

VC++ Example

```
CComBSTR bstrData;
bstrData  = m_altictrl.GetUserData (ulSessionID);
```

VB Example

```
Dim data As String
Data = AltiCtrl1.GetUserData(ulSessionID)
```

SetIVRData

Description

Set IVR data for existing call (used only in Web call). IVR data is the format of "name=value" pairs, which can be used in Web Call and Auto Atendant; for example:

"**FirstName=John&LastName=Anny&Password=hello1234**" and so on.

Note

SetIVRData is a reserved API.

VC++ Syntax

```
BOOL SetIVRData(int SessionID, BSTR string)
```

VB Syntax

```
object.SetIVRData(SessionID,string)
```

Parameters

SessionID [In]: unique ID for a call

String [In]: Set IVR data for a call, maximum length is 256 bytes

Return Type

BOOL - if successful TRUE; otherwise FALSE.

Remark

SessionID - Session ID received from OnCallNotify event needs to be provided as input to this function. This Session ID is valid in ring event and connect event. After Disconnect event, it is no longer valid.

VC++ Example

```
CComBSTR bstrData = _T("ivr data");
m_altictrl. SetURLData (ulSessionID, bstrdata);
```

VB Example

```
Dim data As String
Data = "ivr data"
If (AltiCtrl1.SetURLData(ulSessionID, data) = False) Then
...
Else
...
End If
```

GetIVRData

Description

Get IVR data for existing call.

VC++ Syntax

```
BSTR GetIVRData(int SessionID)
```

VB Syntax

```
object.GetIVRData(SessionID)
```

Parameters

SessionID [In]: unique ID for a call

Return Type

String IVR data for a call; maximum length is 256 bytes.

Remark

SessionID - Session ID received from OnCallNotify event needs to be provided as input to this function. This Session ID is valid in ring event and connect event. After Disconnect event, it is no longer valid.

VC++ Example

```
CComBSTR bstrData;
bstrData = m_altictrl.GetIVRData (ulSessionID);
```

VB Example

```
Dim data As String
Data = AltiCtrl1.GetIVRData(ulSessionID)
```

SetURLData

Description

Set URL data for existing call (used only in Web call). When a call comes to a Web site, it can contain URL data such as "http://www.altigen.com," so that the user application can know where this call comes from.

Note

SetURLData is a reserved API.

VC++ Syntax

```
BOOL SetURLData(int SessionID,BSTR string)
```

VB Syntax

```
object.SetURLData(SessionID,string)
```

Parameters

SessionID [In]: unique ID for a call

String [In] Set URL data for a call; maximum length is 256 bytes.

Return Type

BOOL - if successful TRUE; otherwise FALSE

Remark

SessionID - Session ID received from OnCallNotify event needs to be provided as input to this function. This Session ID is valid in ring event and connect event. After Disconnect event, it is no longer valid.

VC++ Example

```
CComBSTR bstrData = _T("url data");
m_altictrl. SetURLData (ulSessionID, bstrData);
```

VB Example

```
Dim data As String
Data = "url data"
If (AltiCtrl1.SetURLData(ulSessionID, data) = False) Then
...
Else
...
End If
```

GetURLData

Description

Get URL data for existing call

VC++ Syntax

```
BSTR GetURLData(int SessionID)
```

VB Syntax

```
object.GetURLData(SessionID)
```

Parameters

SessionID [In]: unique ID for a call

Return Type

String URL data for a call; maximum length is 256 bytes.

Remark

SessionID - Session ID received from OnCallNotify event needs to be provided as input to this function. This Session ID is valid in ring event and connect event. After Disconnect event, it is no longer valid.

VC++ Example

```
CComBSTR bstrData;
bstrData  = m_altictrl.GetURLData (ulSessionID);
```

VB Example

```
Dim data As String
Data = AltiCtrl1.GetURLData(ulSessionID)
```

SetCallerName

Description

Set Caller Name for existing call

VC++ Syntax

```
BOOL SetCallerName(int SessionID ,BSTR string)
```

VB Syntax

```
object.SetCallerNameData(SessionID,string)
```

Parameters

SessionID [In]: unique ID for a call

String [In] Set URL data for a call, max-length is 256 bytes

Return type

BOOL - The value is TRUE if successful; otherwise FALSE

Remark

SessionID - Session ID received from OnCallNotify event needs to be provided as input to this function. This Session ID is valid in ring event and connect event. While after Disconnect event , it is no longer valid.

VC++ Example

```
CComBSTR bstrName = _T("Tom");
m_altictrl.SetCallerName(ulSessionID, bstrName);
```

VB Example

```
Dim data As String
Data = "Tom";
If (AltiCtrl1.SetCallerName(ulSessionID, data) = False) Then
...
Else
...
End If
```

GetExtNumber

Description

Get extension number which AltiAgent had been logged in.

VC++ Syntax

```
BSTR GetExtNumber()
```

VB++ Syntax

```
object.GetExtNumber()
```

Return type

String extension number for the AltiAgent login is 256 bytes.

VC++ Example

```
CComBSTR bstrData = m_altictrl.GetExtNumber();
```

VB Example

```
Dim data as String
data = AltiCtrl1.GetExtNumber()
```

MakeCall

Description

Make a call through AltiAgent.

VC++ Syntax

```
long MakeCall(BSTR bstrDialNumber);
```

VB++ Syntax

```
object.MakeCall(string)
```

Parameters

```
bstrDialNumber [In]: Destination telephone number, less than 32
digits
```

Return type

0 - Make call command is sent successfully

1 - Unknown error occurs

2 - Disconnect AltiAgent

999 - Exception occurred

VC++ Example

```
BOOL bstart= m_altictrl. MakeCall(L"101");
```

VB Example

```
Dim Result as Integer
Dim dirNum As String
dirNum = "101"
Result = AltiCtrl1.MakeCall(dirNum)
```

GetTrunkAccessCode

Description

Get default trunk access code set in AltiAgent.

VC++ Syntax

```
BSTR GetTrunkAccessCode ();
```

VB++ Syntax

```
object. GetTrunkAccessCode()
```

Return type

```
Default trunk access code set in AltiAgent
```

VC++ Example

```
CComBSTR bstrTrunkAccess= m_altictrl. GetTrunkAccessCode ();
```

VB Example

```
Dim data As String
data = AltiCtrl1.GetIVRData(ulSessionID)
```

Note: The ctrl will show a window in the client application. To hide the window, use the following code

VC++ Example

```
m_altictrl.ShowWindow(SW_HIDE);
```

VB Sample

```
AltiCtrl1.Visible = TRUE
```

DropCall

Description

```
Drops the current call.
```

VC++ Syntax

```
long DropCall();
```

VB++ Syntax

```
long DropCall();
```

Return type

```
0 - Drop call command is sent successfully
!=  0 - Unknown error occurs
```

VC++ Example

```
long result = m_altictrl.DropCall();
```

VB Example

```
Dim data As Long
data = AltiCtrl1.DropCall()
```

SetBK

Description

```
Sets the background color of AltiCtrl.
```

VC++ Syntax

```
long SetBk(short sRed, short sGreen, short sBlue)
```

VB++ Syntax

```
long SetBk(short sRed, short sGreen, short sBlue)
```

Parameters

```
sRed[in]:RGB's red value of the background color of AltiCtrl.
sGreen[in]: RGB's green value of the background color of AltiCtrl.
sBlue[in]: RGB's blue value of the background color of AltiCtrl.
```

VC++ Example

```
m_altictrl.SetBk (128, 101, 122);
```

VB Example

```
AltiCtrl1.SetBk 128, 101, 122
```